REMARKS

In the Final Office Action, Claims 1-38 were examined. Claims 1-4, 6-15, 17-22, 24-29 and 35-38 stand rejected, Claims 5, 16, 23 and 30 are objected to and Claims 31-34 are allowed. In response to the Final Office Action, Claims 1 and 31 are amended, no claims are cancelled and no claims are added. Applicant respectfully requests reconsideration of pending Claims 1-38 in view of the following remarks.

I. Claims Rejected Under 35 U.S.C. §102

The Examiner rejects Claims 1-4, 6-11, 19-22 and 24-29 under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,799,179 to Ebcioglu et al. ("Ebcioglu"). Applicant respectfully traverses this rejection.

"Anticipation requires the presence in a single prior art reference disclosure of each and every element of the claimed invention, *arranged as in the claim.*" *Lindemann Maschinenfabrik v. American Hoist & Derrick* ("Lindemann"), *730 F.2d 452, 1458 (Fed. Cir. 1994)*(emphasis added). Additionally, each and every element of the claim must be exactly disclosed in the anticipatory reference. *Titanium Metals Corp. of American v. Banner* ("Banner Titanium"), *778 F.2d 775, 777 (Fed. Cir. 1985).*

Regarding Claims 1 and 26, Claims 1 and 26 recite the following claim feature, which is neither taught nor suggested by either Ebcioglu or the references of record:

> bypassing execution of instructions having at least one operand with an associated tag that indicates that the operand is a scratch value. (Emphasis added.)

Conversely, Ebcioglu teaches a technique for:

> minimization of CPU overhead from exceptions from speculative instructions as well as efficient handling of an exception for a speculative instruction which turns out to be in the taken path. In accordance with the teaching of this invention, an exception from a speculative instruction is processed if and only if it would occur in the original sequential program. (col. 3, lines 31-37.) (Emphasis added.)

As defined by Ebcioglu:

> Speculative instructions have an extra bit, referred to as a speculative bit, which is set. A speculative instruction is an instruction which is moved above a conditional jump which determines whether or not the speculative instruction is in the taken path. Non-speculative instructions have the speculative bit reset. (col. 3, lines 40-45.) (Emphasis added.)

As further described by Ebcioglu, Ebcioglu describes techniques for processing and tracking original speculative exceptions and secondary speculative exceptions. To track such information, Ebcioglu describes populating of target registers and exception bits and exception files, which according to Ebcioglu:

allows tracking back to the speculative instruction causing the original speculative exception for its resolution. (col. 4, lines 22-24.)

According to the Examiner, the following passage of Ebcioglu teaches the above recited feature of Claims 1 and 26:

> Exception tracking also discards information for speculative exceptions which turn out to be outside the taken path. Speculative exception resolution is triggered when a non-speculative instruction, which is in the taken path, uses an operand from a register having its exception bit set. Speculative exception resolution includes correcting the exception condition which caused the exception and re-executing the instructions which depend on the results of the instruction causing the speculative exception.
>
> The advantage to this approach is that rather than re-executing all speculative instructions, as does the prior art, only those which depend a) on the instruction causing the exception, and b) are in the taken path, are re-executed. This approach saves CPU processing time. (col. 4, lines 24-37.) (Emphasis added.)

Applicant respectfully submits that the cited passage above only refers to the discarding of information associated with the speculative exceptions, which allows tracking back to the speculative exception that caused the original speculative exception that caused the original speculative exception to enable resolution. Applicant respectfully submits that such speculative instructions, as taught by Ebcioglu, are not bypassed when such instructions have an operand that includes a scratch value, as recited by Claims 1 and 26.

As indicated above, the recited features of Claims 1 and 26 refer to the operand of an instruction, which may include a tag that indicates that the operand is a scratch value. Conversely, as taught by Ebcioglu:

> Each RISC instruction can include an opcode field, a first operand field, a second operand field, and a target register number field. An additional one bit field, referred to herein as a speculative bit field, or more simply as a speculative bit is provided to distinguish speculative RISC instructions from non-speculative RISC instructions as described in greater detail below. (col. 5, lines 1-7.) (Emphasis added.)

Applicant respectfully submits that the one bit field of each RISC instruction, as taught by Ebcioglu, which may be set to identify the speculative load instructions, does not refer to an operand within an associated tag, as recited by Claims 1 and 26. Assuming, arguendo, that a register having an exception bit set, as taught by Ebcioglu (*See*, for example, col. 4, lines 3-22), teaches an operand with an associated tag, as recited by Claims 1 and 26, Ebcioglu fails to teach bypassing instructions, which attempt to use the register having its exception bit set. In fact, as indicated by Ebcioglu:

> Speculative execution resolution is triggered when a non-speculative instruction, which is in the taken path, uses an operand from a register having its exception bit set. (col. 4, lines 26-29.) (Emphasis added.)

As taught by Ebcioglu, speculative instructions, which attempt to use a register having its exception bit set, result in secondary speculative exceptions whereas non-speculative instructions that use an operand from a register having exception bits set result in speculative exception resolution. (*See*, col. 4, lines 13-30.) Applicant respectfully submits that bypassing of either speculative or non-speculative or instructions that attempt to use an operand from a register having its exception bit set, is directly contrary to the teachings of Ebcioglu, which teaches the identification of such instructions to limit re-executing of instructions to only speculative instructions, which depend on the instruction causing the exception, and are in the taken path. (*See*, col. 4, lines 34-36.) Therefore, the Examiner fails to illustrate that each and every element of Claims 1 and 26 is exactly disclosed in Ebcioglu, as required to establish a *prima facie* case of anticipation. Id.

Accordingly, Applicant respectfully submits that the Examiner fails to establish a *prima facie* case of anticipation of Claims 1 and 26, since the Examiner fails to illustrate the disclosure of each and every element of Claims 1 and 26 within Ebcioglu. Id. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claims 1 and 26.

Regarding Claims 2-4 and 6-7, Claims 2-4 and 6-7, based on their dependency from Claim 1, are also patentable over Ebcioglu, as well as the references of record. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claims 2-4 and 6-7.

Regarding Claims 27-29, Claims 27-29, based on their dependency from Claim 26, are also patentable over Ebcioglu, as well as the references of record. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claims 27-29.

Regarding Claim 19, Claim 19 recites the following claim feature, which is neither taught nor suggested by either Ebcioglu or the references of record:

> bypassing execution of instructions having at least one operand with an associated tag that indicates that the operand is a scratch value.

As indicated above, Ebcioglu teaches that either a secondary speculative exception or a speculative exception resolution occurs when an instruction uses an operand from a register having its exception bit set. Hence, Applicant respectfully submits that Ebcioglu fails to teach bypassing instructions that use an operand from a register having its exception bit set.

Consequently, the Examiner is prohibited from establishing the disclosure of bypassing executions of instructions having at least one operand with an associated tag that indicates that the operand is a scratch value, as recited by Claim 19, as required to establish a *prima facie* case of anticipation. Id. Therefore, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claim 19.

Regarding Claims 20-22 and 24-25, Claims 20-22 and 24-25, based on their dependency from Claim 19, are also patentable over Ebcioglu, as well as the references of record. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claims 20-22 and 24-25.

The Examiner rejects Claims 35-38 under 35 U.S.C. §102(b) as being clearly anticipated by U.S. Patent No. 5,933,627 to Parady ("Parady"). Applicant respectfully traverses this rejection.

Regarding Claim 35, Claim 35 recites the following claim features, which are neither taught nor suggested by either Parady or the references of record:

> setting a tag of a register when an instruction having the register as a destination results in a cache miss, to identify the register as containing a scratch value;
>
> bypassing execution of instructions having at least one operand with an associated tag that indicates that the operand is a scratch value until at least one instruction is detected that results in a cache miss. (Emphasis added.)

According to the Examiner, the setting of a tag of a register, as recited above, is taught by element 80 of FIG. 2 of Parady. Element 80, as shown in FIG. 2 of Parady, illustrates L2 cache tags, which as known to those skilled in the art are used to determine whether requested data is contained within an L2 cache. The only reference to element 80 shown in FIG. 1 of Parady is the description that L2 cache tag memory 80, and L2 cache data memory 82 are shown in FIG. 2. (*See*, col. 3, lines 27-28.) In contrast to the above recited features of Claim 35, Parady is directed to a method and apparatus for switching between threads of a program in response to a long latency event. As described with Parady:

> The indication that a thread switch is required is provided on a line 114 providing an L2-miss indication from cache control/system interface 22 of FIG. 1. Upon such an indication, a switch to the next thread will be performed, using, in one embodiment, the next thread pointer on line 116. The next thread pointer is two bits indicating the next thread from an instruction which caused the cache miss. (col. 3, lines 58-65.) (Emphasis added.)

As further described by Parady:

> These two bits of the next thread pointer come from a thread field 118 in an instruction 120. Instruction 120 also includes an opcode field 122 and a source and destination register fields 124 and 126, respectively. By adding the two bit thread field 118 to appropriate instructions, control can be maintained over thread switching operations. In one embodiment, the thread field is added to all load and store operations. (col. 3, line 66 - col. 4, line 6.) (Emphasis added.)

Accordingly, as illustrated in FIG. 3 of Parady, thread switching logic 112, in response to an L2 cache miss received over line 114, switches execution to a next thread according to next thread pointer received over line 116. Hence, Parady fails to teach or suggest the setting of a tag of a

register when an instruction having the register as a destination results in a cache miss to identify the register as containing a scratch value, as recited by Claim 35.

Furthermore, Claim 35 limits bypassing of execution to instructions that have at least one operand with an associated tag that indicates that the operand is a scratch value. Conversely, Parady teaches that upon indication of a cache miss, a switch to the next thread is performed using a next thread pointer received on line 16, which may be part of a thread field 118 and an instruction 120, as shown in FIG. 4 of Parady.

Hence, Parady fails to teach the bypassing of execution of instructions, as recited by Claim 35. However, the case law is clear in establishing that an anticipatory reference must disclose each and every element of a claim in order to anticipate the claim. Id. Accordingly, for at least the reasons described above, Applicant respectfully submits that the Examiner fails to illustrate the disclosure of each and every element of Claim 35, as recited above, as required to establish a *prima facie* case of anticipation. Id.

Consequently, Applicants respectfully submit that the Examiner failed to establish a *prima facie* case of anticipation of Claim 35, since the reference cited by the Examiner does not teach or suggest each of the recited features of Claim 35. Id. Therefore, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claim 35.

Regarding Claims 36-38, Claims 36-38, based on their dependency from Claim 35, are also patentable over Parady, as well as the references of record. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §102(b) rejection of Claims 36-38.

IV. Claims Rejected Under 35 U.S.C. §103

The Examiner rejects Claims 12-15, 17 and 18 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,651,125 issued to Witt et al. ("Witt") in view of Ebcioglu. Applicant respectfully traverses this rejection.

To establish a *prima facie* case of obviousness, the following criteria must be met: (1) there must be some suggestion or motivation to modify the reference or combine the reference teachings, (2) there must be a reasonable expectation of success, and (3) the prior art references must teach or suggest all the claim limitations. (MPEP §2142)

Regarding Claim 12, Claim 12 recites the following claim features, which are neither taught nor suggested by the combination of Witt in view of Ebcioglu:

> an execution engine having a plurality of instructions which when executed cause the processor to perform actions including: ...
> bypassing execution of instructions having at least one operand with an associated tag that indicates that the operand is a scratch value.

As indicated above, Ebcioglu teaches away from bypassing execution of instructions, which have an operand that indicates that the operand is a scratch value. Ebcioglu teaches that speculative

instructions, which attempt to use a register having its exception bit set, results in secondary speculative exceptions, which require storage of information to allow tracking back to the speculative instruction that caused the original speculative exception resolution. (*See*, col. 4, lines 13-24.)

As further taught by Ebcioglu, speculative exception resolution is triggered when a non-speculative instruction, which is in the taken path, uses an operand from a register having its exception bit set. (*See*, col. 4, lines 26-29.) Applicant respectfully submits that bypassing of such instructions would render Ebcioglu unsatisfactory for its intended purpose of limiting re-execution of speculative instructions to only those instructions, which depend on the instruction causing the exception and that are on the taken path. (*See*, col. 4, lines 33-36.) Hence, Applicant respectfully submits that Ebcioglu teaches away from bypassing execution of instructions that have an operand indicating that the operand contains a scratch value, as recited by Claim 12.

Consequently, Applicant respectfully submits that the Examiner fails to establish that the combination of Witt in view of Ebcioglu teaches or suggests each of the above recited features of Claim 12, as required to establish a *prima facie* case of obviousness. Consequently, Claim 12 is patentable over the combination of Witt in view of Ebcioglu, as well as the references of record. Consequently, Applicant respectfully requests that the Examiner reconsider and withdraw the §103(a) rejection of Claim 12.

Regarding Claims 13-15, 17 and 18, Claims 13-15, 17 and 18, based on their dependency from Claim 12, are also patentable over the combination of Witt in view of Ebcioglu, as well as the references of record. Therefore, Applicant respectfully requests that the Examiner reconsider and withdraw the §103(a) rejection of Claims 13-15, 17 and 18.

V. Allowable Subject Matter

The Examiner has objected to Claims 5, 16, 23 and 30 as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form, including all of the limitations of the base claim and any intervening claims. However, Claims 5, 16, 23 and 30 are also patentable based on their dependency from Claim 1. Accordingly, Applicant requests that the Examiner allow Claims 5, 16, 23 and 30.

The Examiner has indicated that Claims 31-34 are allowed. Applicant respectfully thanks the Examiner for recognizing the allowability of Claims 31-34, as well as the allowability of Claims 5, 16, 23 and 30 if rewritten in independent format.

CONCLUSION

In view of the foregoing, it is submitted that Claims 1-38 patentably define the subject invention over the cited references of record, and are in condition for allowance and such action is earnestly solicited at the earliest possible date. If the Examiner believes a telephone conference would be useful in moving the case forward, he is encouraged to contact the undersigned at (310) 207-3800.

If necessary, the Commissioner is hereby authorized in this, concurrent and future replies, to charge payment or credit any overpayment to Deposit Account No. 02-2666 for any additional fees required under 37 C.F.R. §§1.16 or 1.17, particularly, extension of time fees.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR, & ZAFMAN LLP

Dated: January 5, 2005

By: _____
Joseph Lutz, Reg. No. 43,765

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025
(310) 207-3800

CERTIFICATE OF MAILING:
I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail, with sufficient postage, in an envelope addressed to: Mail Stop AF, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on January 5, 2005

_____
Marilyn Bass January 5, 2005